Software Tools for Ensuring Computational Reproducibility

Yuan, Daniel S.
Johns Hopkins University School of Medicine, Baltimore, MD, USA

Reproducibility is a cardinal tenet of experimental science. Although the importance of experimental reproducibility is unquestioned, little has been written about the equal importance of computational reproducibility. The ability to reproduce computed results is vital if collaborative research with open access to shared datasets is to be meaningful. This poster discusses software tools that ensure computational reproducibility by describing each step of a computation as a closed system. Inputs and outputs are documented by checksums, while the function or method calls that transform inputs into outputs are documented by executable code. Data validation tools are included, and error-trapping and database tools ensure that data processing can be robustly performed and rigorously documented for many data objects at a time in the face of unanticipated errors and very large datasets. Thus, documentation is generated on-the-fly rather than as an afterthought, and data processing is made "failsafe". "failsafe" is a freestanding module written in the Python programming language and will be distributed through established Open Source channels. Applications to date include management of a self-adjusting primer picker on an IRIX system and ad hoc data processing in Linux. "failsafe" programming will provide a much needed infrastructure for projects where intricate and evolving programs must process thousands of datasets in an openly documented and consistent manner, as in an NIH-supported, microarray-based yeast functional genomics project currently underway at this institution.

Supported by the Burroughs-Wellcome Program in Computational Biology at Johns Hopkins and by NIH grant P01CA16519.